CASIO PB-100

取扱説明書

カシオ計算機株式会社

はじめに

このたびは《**カシオPB-100**》をお買い上げいただきまして、誠にありがとうございます。

本機は、これからコンピュータを習おうとする方に最適な〈ハンディータイプパーソナルコンピュータ〉です。

このハンディーパソコンにより、コンピュータの世界を把握し、BASIC言語によるプログラミングを自在に行なうことができます。

> この取扱説明書は、計算機の基本的な操作およびBASIC言語の概要について説明してあります。ご使用前によくお読みいただき、各機能を十二分にマスターの上、ご使用上の注意を守って、末ながくご使用ください。

製品裏面の銘板にありますBM(BMマーク)は、電卓の品質向上とアフターサービス体制の確立のため、(社)日本事務機械工業会が品質の認定試験基準を定め、これに合格した電卓だけに認定の証として許可されるマークです。

目　　次

ご使用の前に

この計算機は、カシオの高度な電子技術と品質管理のもとで、厳重な検査工程を経て、皆様のお手もとに届けられています。

本機を末ながくご愛用いただくために、次の点にご留意のうえ、お取り扱いください。

■ご使用上の注意

- 計算機は精密な電子部品で構成されていますので、絶対に分解しないでください。
  また、投げたり落したり等のショックや、急激な温度変化を与えないでください。
  特に、高温の所、湿気やホコリの多い所に放置したり保管することはしないでください。
  なお、温度が低いときは表示の応答速度が遅くなったり、点灯しなくなることがありますが、通常の温度になると正常にもどります。
- アダプター差し込み口には、本機専用のオプション以外は接続しないでください。
- 計算機の演算中は〝ー〟を表示し、この間のキー操作は一部キーを除いて無効ですから、常に表示を確認しながら、確実にキーを押してください。
- 電池は、計算機を使わない場合でも2年に1度は交換してください。
  特に消耗ずみの電池を放置しておきますと、液もれをおこし、故障等の原因になりますので、計算機内には絶対に残しておかないでください。
- 計算機のお手入れは、シンナー・ベンジン等の揮発性液体をさけ、「乾いた柔らかい布」あるいは、「中性洗剤液に浸し固くしぼった布」でおふきください。

■保証・アフターサービス

- 保証は、別紙の保証書の内容によりますので、よくお読みのうえ、記入事項を確認して、大切に保管してください。
- 万一故障したときは

  ❶お買い上げ店　❷カシオ計算機サービスセンター

  のうち、ご都合のよい所へ、必ず保証書をそえて、ご持参またはご郵送ください。
  この場合、故障内容を具体的にお知らせください。
- 修理依頼される前には、この説明書をもう一度お読みになると共に、電源の状態および、プログラムミス、操作ミスがないかをよくお調べください。
- ご不明の点やご質問、お問い合わせ等は、67ページのカシオ計算機へ直接ご連絡ください。

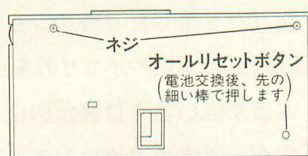
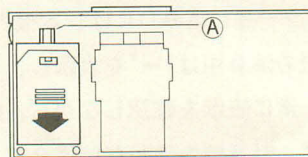
電源および電池交換について

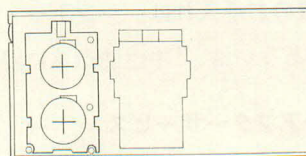
本機は、リチウム電池《CR-2032》2個を電源とします。

コントラストのボリウム(10ページ参照)を最も濃くなる位置にしても表示が薄く見えるときは電池が消耗していますので、早目に下記に示すように電池交換をしてください。

なお、計算機が正常に使用できても、2年に1度は必ず電池交換をしてください。

■電池交換の仕方

①電源スイッチを切ってから、裏面の2個のネジをはずし、裏ブタをはずします。

②図のⒶを押しながら矢印方向にスライドさせて、電池押さえ板をはずします。

③古い電池を2個とも取り出します。
(電池ボックスを下に向けて軽くたたけば簡単にはずれます。)

④新しい電池の表面を乾いた布でよくふいてから㊉側を上にして入れます。

⑤電池押さえ板を電池を押さえながらスライドさせてとじます。

⑥裏ブタをネジ止めし、電源スイッチをONにしてからリセットボタンを先の細い棒で押してください。

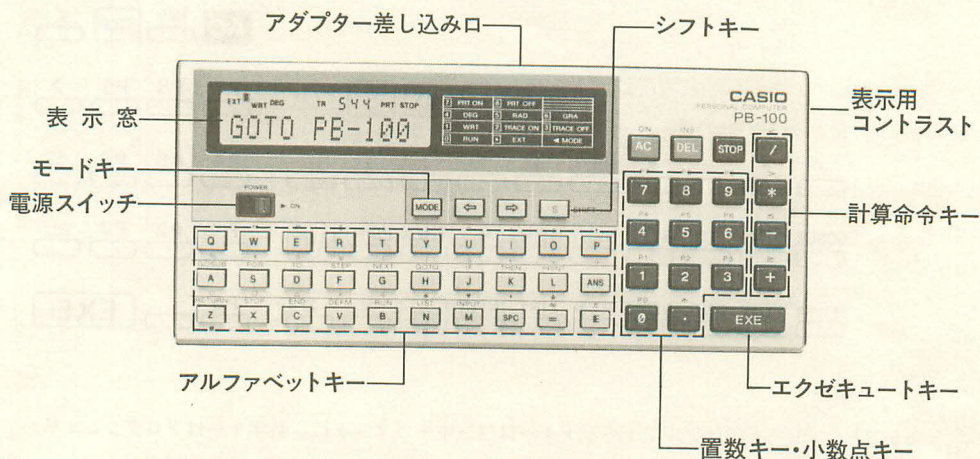
※電池は必ず2個とも交換してください。

※消耗ずみの電池は絶対に火中に投入しないでください。破裂することがあり、非常に危険です。

※電池の㊉㊀は絶対にまちがえないようにしてください。

電池は、幼児の手のとどかないところに保管してください。

万一、飲み込んだ場合にはただちに医師と相談してください。

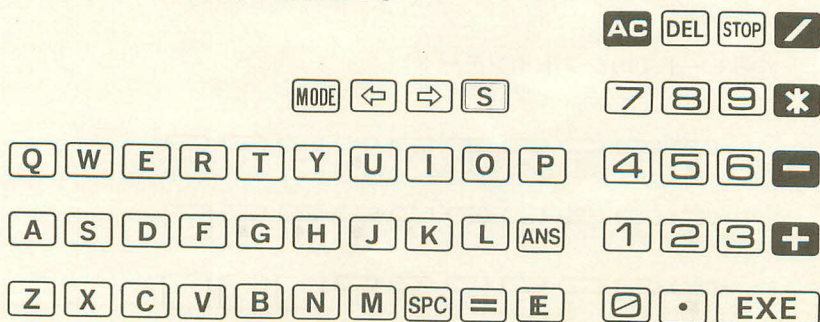
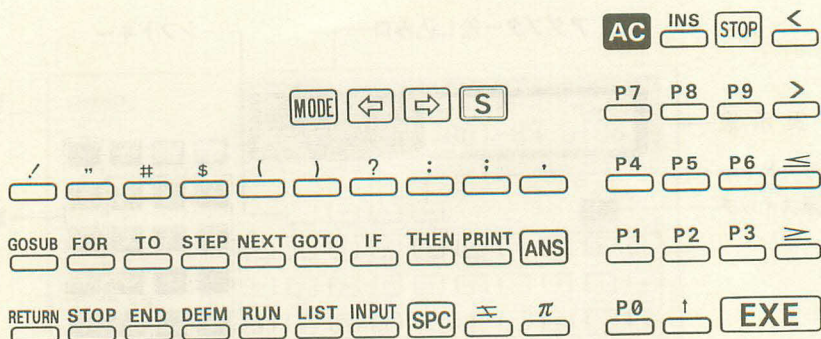
各部の名称とそのはたらき

■各部の名称

各キーは１つまたは２つの働きを持っています。この働きは、直接押すシフトアウトモードと、SHIFT（シフト）キーに続いて押すシフトインモードにより使いわけます。

〈 例 〉

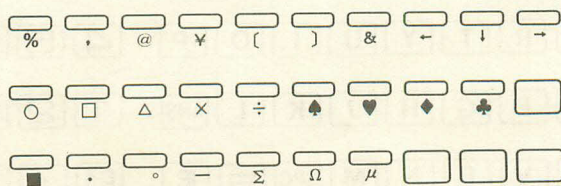
GOSUB ――シフトインモード

A ――シフトアウトモード

シフトアウトモードでのキーのはたらき

シフトインモードでのキーのはたらき

※シフトインモードではアルファベットキーはワンキーコマンドに、数字キーはプログラムエリア指定となります。

★アルファベットキーは、シフトアウトモードとシフトインモードの他に拡張モード(MODE・と押す。"EXT"表示)で、英小文字・特殊記号を表示します。

拡張モードでのシフトアウトモード

拡張モードでのシフトインモード

S シフトキー

このキーを押しますと、シフトインモードとなり("S"を表示)、キーボード上のシフトイン機能が使えます。(本書ではSHIFTと記します。)

MODE モードキー

計算機の状態および角度単位をあらかじめ指定するとき・0〜8の数字キーと合わせて押します。

MODE・……"EXT"を表示し、拡張モードとなり、英小文字・特殊記号が使えます。
MODE0……"RUN"を表示し、マニュアル計算およびプログラム計算が行なえます。
MODE1……"WRT"を表示し、プログラムの書き込みおよびチェック、編集が行なえます。
MODE2……"TR"を表示し、実行トレースが行なえます。(詳しくは43ページ)
MODE3……"TR"が表示している場合は"TR"が消え、実行トレース機能が解除されます。
MODE4……"DEG"を表示し、角度の単位を〈度〉に指定します。
MODE5……"RAD"を表示し、角度の単位を〈ラジアン〉に指定します。
MODE6……"GRA"を表示し、角度の単位を〈グラジアン〉に指定します。
MODE7……"PRT"を表示し、プリンタ接続時はプリント出力をすることができます。
MODE8……"PRT"が表示している場合は"PRT"が消え、プリント出力が解除されます。

⇦⇨ カーソルキー

カーソルの位置を左右に移動したいとき押します。1回押すと一文字分移動し、押し続けると自動的にリピートします。

ON/AC オールクリヤーキー

- 表示をすべてクリヤーにしたいとき押します。
- プログラム実行中に押された場合はプログラムの実行を中止させます。
- エラーメッセージが表示された場合は、エラーを解除するために押します。
- オートパワーオフ(自動節電機能…12ページ参照)が働いて、表示が消えているときに電源をオンにするために押します。

INS/DEL デリート/インサートキー

- カーソルが点滅している所の入力文字を1文字分削除します。
- シフトインモードでは、挿入のため1文字分あけるために押します。

STOP ストップキー

プログラム実行中に押しますと、"STOP"を表示し、一行の区切りでプログラムの実行を停止させます。
実行トレース中で"STOP"を表示しているときに押しますと、プログラムエリアNo.と行番号を表示します。

EXE エクゼキュートキー

- マニュアル計算の答を求めるとき、「=」のかわりに押します。
- "WRT"モードでは、プログラムを書き込むときに、行番号ごとの計算機への書き込み(記憶)として押します。このキーを押しませんと書き込まれません。
- "RUN"モードでは、プログラムの実行中のデータ入力のためや、"STOP"表示中にプログラムの実行を継続するために押します。

ANS アンサーキー

マニュアル計算やプログラム計算で直前に行なわれた計算結果(答)を呼び戻したいとき押します。

[E] エキスポーネント(指数部置数)/パイキー

・指数部を置数するとき、仮数部の置数後に押します。

〈例〉 2.56×10^{34} ⟶ 2·56 [E] 34

※指数部は最高±99までで、それ以上はエラーとなります。

[=] イコールキー/比較キー

・代入文のときやIF文中の比較(等号)のとき押します。

・シフトインモードでは、IF文中の比較のとき押します。

[7][8][9][4][5][6][1][2][3][0][・] 置数キー/プログラムNo.キー

・数値を計算機に入力する(置数する)とき押します。[・]は小数点の位置で押します。

・シフトインモードでは[P0]~[P9]はプログラムNo.の指定となり、プログラムが書き込まれているときはプログラムがスタートされます。

・[・]キーは、シフトインモードではべき乗(x^y)を求めるとき押します。

[+][−][*][/] 計算命令キー/比較キー

・四則演算をするとき、それぞれの位置で押します。

※[*]は"×"(乗算)のとき、[/]は"÷"(除算)のとき押します。

・シフトインモードでは、IF文中の判断のための比較として書くとき押します。

!	"	#	$	(	)	?	:	;	,
Q	W	E	R	T	Y	U	I	O	P

GOSUB	FOR	TO	STEP	NEXT	GOTO	IF	THEN	PRINT
A	S	D	F	G	H	J	K	L

RETURN	STOP	END	DEFM	RUN	LIST	INPUT	
Z	X	C	V	B	N	M	SPC

アルファベットキー/ワンキーコマンド・キャラクターキー

- プログラムを書き込むときや、コマンド・関数命令を書くとき、これ等のキーを押せばアルファベットが表示されます。SPCはスペース(空白)を開けるとき押します。
- Q〜Pは、シフトインモードでキーパネル上に書かれているキャラクターが表示されます。
- A〜Mは、シフトインモードでキーパネル上に書かれているワンキーコマンドを使うとき押します。

表示の見方

```
EXT Ⓢ RUN WRT DEGRADGRA TR   544   PRT STOP
-1.23456789_
```

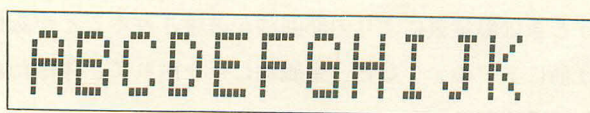
計算数値や答を表示します。それぞれの桁が横5×縦7ドットで構成され、数値や文字を最大12桁まで表示します。(数字0はØと表示します)式や文が12桁をこえた場合は数字や文字が左に送られ、最大62文字が入力できます。

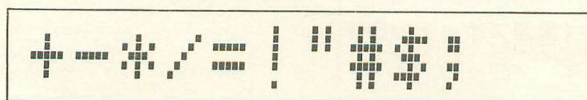
カーソルは入力文字が55文字までは"_"の点滅となり、56文字以上になると"■"の点滅となります。

表示上部の4桁は残りのステップ数を表示します。

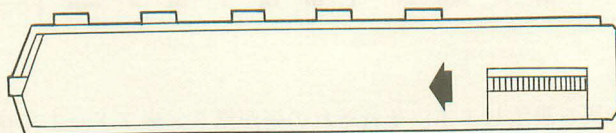
なお、演算中は上部4桁の一番右に"-"が、また、"DEG""RAD""GRA"の角度単位や"S"(SHIFTキーを押したとき)、"RUN"(RUNモード)、"WRT"(WRTモード)、"TR"(トレースモード)、"PRT"(PRTモード)、"STOP"などの記号も、それぞれの状態のとき表示します。

●アルファベット表示例

```
ABCDEFGHIJK
```

●記号表示例

```
+-*/=!"#$;
```

計算をはじめる前に

■コントラスト調整

表示のコントラスト(濃度)の調整は、本体右側にある調整用ボリウムで行ないます。

矢印方向に回すと表示が濃くなり、反対に回すと薄くなります。これは、電池容量による表示の濃淡の調整や、角度による見易さを調整します。

■増設RAMパック(別売)について

本機のRAMエリア(メモリー)は標準で544ステップ、26メモリーですが、別売のRAMパックを増設することにより、最大1,568ステップに増やすことができます。
この増設されたRAMエリアは、標準のエリアと同様に使え、ステップ数の増量や、メモリーの増設(11ページ参照)を行なうことができます。

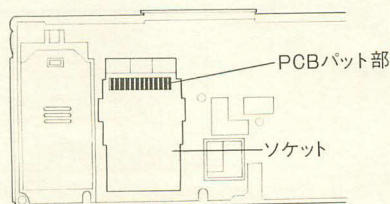
●RAMパックの取り付け方

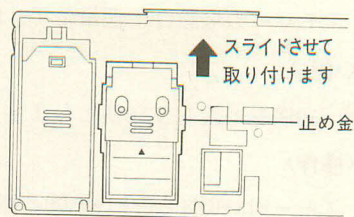
〈準備〉

パックを取り扱うときは静電気により内部回路が破壊されることがありますのでパックを取り扱う前にドアのノブなど、金属物に手を触れて、体にたまった静電気を放電させておいてください。

〈手順〉

(1)電源を切ります。(電源スイッチ→OFF)

(2)本体裏側のネジを２本ともはずし、裏ブタをはずします。

(3)RAMパックについてる止め金を下にさげて、パックを本体のソケットに入れ、止め金を押さえながらスライドさせます。

※RAMパックのコネクター部、PCBパット部には絶対手を触れないでください。

(4)裏ブタをネジ止めします。

★RAMパックを本体に増設したり、取りはずしたりした後は、必ず電源スイッチを入れた後に本体裏面のオールリセットボタンを先の細い棒などで押してください。オールリセットボタンを押さない場合は、メモリー内容が変化したり、無意味な表示が表われることがあります。

★パックのコネクターおよびPCBパット部にゴミ、ホコリ、指紋などがつきますと接続不良の原因となりますので、絶体に触れないようにしてください。

★取りはずしたパックは必ずケースに入れ、ゴミ、ホコリのない所に保管してください。

■メモリーの増設

メモリー(変数)は通常26メモリーで、そのときのステップ数は544ステップとなります。

このメモリーは、標準で最大94個、RAMパック増設時で最大222個まで増設することができます。このメモリーの増設は、1メモリーにつき8ステップの換算でプログラムステップをメモリーに交換します。

メモリー数		26個	27個	28個	～	46個	～	94個	～	200個	～	222個
プログラムステップ数	標準時	544ステップ	536ステップ	528ステップ	～	384ステップ	～	0ステップ	～	－		－
	増設時	1568ステップ	1560ステップ	1552ステップ	～	1408ステップ	～	1024ステップ	～	176ステップ	～	0ステップ

メモリーの増設は1個単位で、DEFMコマンドを使い行ないます。

〈 例 〉

30個を増設して56個とします。

〈操作〉

モードはRUNモード(MODE 0と押す)またはWRTモード(MODE 1と押す)で行ないます。

DEFM　30 EXE　　　＊＊＊VAR：56

※DEFMはD E F Mと押してもSHIFT DEFMと押しても同じです。

現在メモリー数がいくつに設定されているかを確認するときもDEFMコマンドを使います。

〈 例 〉

メモリー数が合計で56個に設定されています。

DEFM　EXE　　　＊＊＊VAR：56

★すでに相当数のプログラムステップが使用されている場合には、書き込みずみのプログラムを保護するために、ステップ数不足となる指定はエラー(ERR1…ステップ数不足)となります。

★専用文字変数($)は、特別メモリーのため、指定時には数えません。

■オートパワーオフ(自動電源OFF)

スイッチの切り忘れによるムダな電力消費を防ぐ自動節電機能で、操作完了後(プログラム計算中を除く)約7分で自動的に電源オフになります。

この場合は、電源スイッチを入れ直すか、ACキーを押せば、再び電源オンになります。

※電源オフになってもメモリー内容およびプログラム内容は消えませんが、角度指定や各モード指定("WRT"、"TR"、"PRT"等)はすべて解除されます。

計算の仕方

マニュアル計算およびプログラム計算の実行は〝RUN〞モード(〝RUN〞表示＝MODE 0)で行なってください。

なお、〝DEG〞〝RAD〞〝GRA〞の角度単位は、角度に関係ない計算の場合は何が表示されていても関係ありません。

■ 計算の優先順位

本機は「計算の優先順位」を計算機自身が判別し、その順位に従って演算を行ないます。

計算の優先順位は次のように定められています。

①関数(SIN, COS, TAN等)

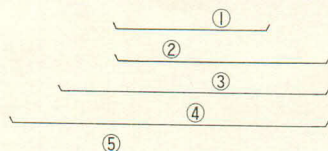
②べき乗

③×，÷(＊，／)

④＋，－

優先順位が同じときは頭から(左から)計算します。
なお、カッコが使用されたときは、カッコ内が最優先されます。

〈例〉

2＋3＊SIN (17＋13)↑2＝2.75

■ 入出力桁数と演算桁数

本機の入力桁数は仮数部12桁、指数部2桁で、内部演算も仮数部12桁、指数部2桁で行なわれます。

この範囲は1×10^{-99}～$\pm9.99999999999\times^{+99}$および0です。

出力桁数は仮数部10桁ですが、指数部が付く場合は仮数部8桁、指数部2桁となります。

※関数の答等で、表示桁数(12桁)をこえる場合は0と小数点および負号を含めて12桁まで表示されます。

〈例〉

$(1\times10^5)\div7=14285.71429$	1 E 5 ÷ 7 EXE	14285.71429
$(1\times10^5)\div7-14285=0.7142857$	1 E 5 ÷ 7 － 14285 EXE	0.7142857

計算の結果(答)が10^{10}(100億)以上または10^{-3}未満のときは、自動的に指数表示になります。

〈例〉

1234567890×10=12345678900　　1234567890 ＊ 10 EXE　　1.2345678E10

(=1.2345678×10^{10})　　指数サイン

※指数部は、仮数部の後に指数サインと共に表示されます。

1.234÷10000=0.0001234　　1.234 / 10000 EXE　　1.234E-04

(=1.234×10^{-4})

■ 基本的な計算の仕方

(1)計算符号と関数命令

BASICに使われる演算符号のうち、四則計算用では"+"と"−"は一般の計算式に使われるものと同じですが"×"と"÷"は"*"と"/"を使います。

〈例〉

2+3−4×5÷6　という計算式は

2+3−4*5/6　となります。

また、本機で使える組込関数は次の通りです。

関数名		書式
三角関数	$\sin x$	SIN x
	$\cos x$	COS x
	$\tan x$	TAN x
逆三角関数	$\sin^{-1} x$	ASN x
	$\cos^{-1} x$	ACS x
	$\tan^{-1} x$	ATN x
平方根	$\sqrt{x}$	SQR x
指数関数	e^x	EXP 1*
自然対数	$\ln x$	LN x
常用対数	$\log x$	LOG x
整数化	INT x	INT x
整数部除去	FRAC x	FRAC x
絶体値化	$\|x\|$	ABS x

符号化　正数→1　　　SGN x
　　　　0→0
　　　　負数→－1

四捨五入 [xの10^yを四捨五入]　RND (x, y)※

乱　数　　　RAN #

※RND関数は必ず引数を(　)でくくります。

★EXPは指数表の数値を呼び出す命令です。

■ 直前の計算結果の呼び出し

マニュアル計算およびプログラム計算実行後の計算結果は、次の計算が実行されるまで記憶されています。この結果は[ANS]キーにより表示されます。

〈例〉

741＋852＝1593

2431－1593＝838

〈操作〉

キー操作	表示
[7][4][1][+][8][5][2]	741+852
[EXE]	1593
[2][4][3][1][−][ANS]	2431−1593
[EXE]	838

また、計算直後に表示された数値は、そのまま次の計算にも使えます。

〈例〉

25.3＋13.9＝39.2

39.2×7.6＝297.92

キー操作	表示
[2][5][・][3][+][1][3][・][9][EXE]	39.2
[*][7][・][6]	39.2*7.6
[EXE]	297.92

■ エラーメッセージ

数式や代入文が、BASIC言語の記法上正しくないときや、計算機が計算範囲をこえて使用されると、実行時にエラーとなり、エラーメッセージが表示されます。べき乗($x \uparrow y$)計算の場合は、$x < 0$でもyが整数であればエラーメッセージは表示されません。

このエラーメッセージはマニュアル計算中は

表示	意味
ERR2	(構文エラー)
ERR3	(数学的エラー)

プログラム計算中は

表示	意味
ERR2 P0-10	P0のプログラムの10行目に構文エラー発生
ERR3 P2-30	P2のプログラムの30行目に数学的エラー発生

と表示されます。(エラーメッセージの意味は62ページを参照してください)

※演算結果が$\pm 9.99999999999 \times 10^{+99}$をこえた場合はオーバーフローとなり、エラーメッセージ(ERR3)が表示されます。また、1.0×10^{-99}未満のときはアンダーフローとなり、演算結果は0となります。

■ キー操作

マニュアル計算の場合でも、プログラム計算およびプログラムの書き込みの場合でも、キー操作は次のように行ないます。

(1)キー入力

●アルファベット(英文字)入力

〈 例 〉ABCを入力する。

〈操作〉[A][B][C]　　ABC

〈 例 〉SINを入力する。

〈操作〉[S][I][N]　　SIN

●数字入力

〈 例 〉123を入力する。

〈操作〉[1][2][3]　　123

〈 例 〉96.3を入力する。

〈操作〉[9][6][・][3]　　96.3

● 記号入力

〈 例 〉 $#? を入力する。

〈操作〉 [SHIFT][$/R][SHIFT][#/E][SHIFT][?/U]

$#?

〈 例 〉 @¥Ωを入力する。

〈操作〉 [MODE][・]……(拡張モード指定)

[SHIFT][E/@][SHIFT][R/¥][SHIFT][N/Ω]

[MODE][・]……(拡張モード解除)

拡張モード

EXT
EXT @¥Ω
@¥Ω

● 指数付数値の入力

〈 例 〉 7.896×10^{15} を入力する。

〈操作〉 [7][・][8][9][6][E][1][5]

7.896E15

〈 例 〉 -2.369×10^{-45} を入力する。

〈操作〉 [−][2][・][3][6][9][E][−][4][5]

−2.369E−45

⑵入力内容の変更(訂正、削除、挿入)

● 訂正

訂正は、訂正したい箇所にカーソルを移動し([⇦][⇨]を使う)、その位置で新しく訂正したい文字や数字・記号のキーを押します。

〈 例 〉 "A$" を "B$" に訂正する。

A$_

〈操作〉 カーソルを左に2文字分移動させる。

[⇦][⇦]

[B]キーを押す。

A$
B$

〈 例 〉 "LIST" を "RUN" に訂正する。

LIST_

〈操作〉 カーソルを左に4文字分移動させる。

[⇦][⇦][⇦][⇦]

[R][U][N][SPC] または [SHIFT][RUN/B] と押す。

LIST
RUN _

● 削除

削除は、削除したい箇所にカーソルを移動させ、[DEL]キーを押します。

1回押すごとに1文字分削除され、削除された文字分だけ左につめられます。

〈 例 〉 "SIIN" と書いたものから "I" を1文字分削除する。

SIIN_

〈操作〉 カーソルを2文字分移動させる。

[⇦][⇦]

SIIN

[DEL]を押す。

SIN

〈 例 〉 "INPUT X,Y" から "X," を削除する。

INPUT X,Y_

〈操作〉 カーソルを3文字分左に移動させる。

[⇦][⇦][⇦]

INPUT X,Y

[DEL][DEL]と押して2文字分削除する。

INPUT Y

● 挿入

挿入は、挿入したい箇所の右の文字にカーソルを移動させ、その位置で[SHIFT][INS]と押せば1文字分あきますので、その箇所に挿入したい文字や数字・記号のキーを押します。

〈 例 〉 "T=A$" と書いたものを "T$=A$" と直す。

T=A$_

〈操作〉 カーソルを左に3文字分移動させる。

[⇦][⇦][⇦]

T=A$

[SHIFT][INS]と押して、1文字分あける。

T_=A$

挿入したい文字[SHIFT][$]と押す。

T$=A$

〈 例 〉 "PRINT X" を "PRINT SIN X" に直す。

PRINT X_

〈操作〉 カーソルを左に1文字移動させます。

[⇦]

PRINT X

"SIN" を入れるために3文字分あける。

[SHIFT][INS][SHIFT][INS][SHIFT][INS]

PRINT _ X

[S][I][N]と押す。

PRINT SINX

以上が入力内容の変更の方法です。

マニュアル計算

■ マニュアル計算とは

プログラムとして計算式を記憶させて、計算機に自動的に計算させるのではなく、数式の右辺の計算を左辺に代入したり、計算式を手計算により行なったり、変数の内容を呼び出したりする操作を行なうことをマニュアル計算と呼びます。

■ マニュアル計算の操作方法

●四則計算は完全数式通りに、+ - *（×）/（÷）とEXE（＝）を使います。EXEキーは計算結果を求めるキーで、"＝"の働きをします。

〈 例 〉 12＋36－9×5÷4＝36.75

〈操作〉 1 2 + 3 6 - 9 * 5 / 4 EXE

```
12+36-9*5/4
36.75
```

●関数計算は四則計算を含めて完全数式通りで、関数コマンドの後にデータを書き込みます。

〈 例 〉 log 1.23＝0.0899051114

〈操作〉 LOG 1.23 EXE

```
LOG1.23
0.0899051114
```

※アルファベットと数字はこの説明書では枠をはずして記します。

例：S I N (1 5 + 8) EXE → SIN SHIFT (15 + 8 SHIFT) EXE

●メモリー計算などのように、数値や計算結果を記憶させたり、合計をとる場合には変数を使います。この変数は英文字(A,B,～Z)または英文字と数字(0,1,2,～9)を組み合わせたもの(配列として用いた場合)を使い、記憶用のメモリーの働きをします。

変数に数値や計算結果を入れるには、代入式を使います。

〈 例 〉 1234を変数Aに記憶させる。

〈操作〉 A = 1234 EXE

```
A=1234
_
```

〈 例 〉23×56の答を変数Kに加える。

〈操作〉 K[=]K[+]23[*]56 [EXE]

K=K+23*56
_

この方法は、プログラム中の代入文と同じ操作をマニュアルで行なう方法です。

※[EXE]キーを押す前の訂正はカーソルを訂正箇所に合わせ、正しいキーを押します。（17ページ参照）

※表示全部を消す場合は[AC]を押します。

■ マニュアル計算例

基本計算の仕方

加減乗除計算

〈 例 〉23＋4.5－53＝－25.5

〈操作〉 23[+]4.5[−]53[EXE]　　　-25.5

〈 例 〉56×(－12)÷(－2.5)＝268.8

〈操作〉 56[*][SHIFT]([−]12[SHIFT])[/][SHIFT]([−]2.5[SHIFT])[EXE]　　　268.8

〈 例 〉12369×7532×74103＝6.9036806×10^{12}
(＝6903680600000)

〈操作〉 12369[*]7532[*]74103[EXE]　　　6.9036806E12

〈 例 〉1.23÷90÷45.6＝2.9970760×10^{-4}
(＝0.00029970760)

〈操作〉 1.23[/]90[/]45.6[EXE]　　　2.9970760E-04

※答が10^{10}(100億)以上および10^{-3}(0.001)未満のときは指数表示となります。

〈 例 〉7×8＋4×5＝76

〈操作〉 7[*]8[+]4[*]5[EXE]　　　76

〈 例 〉12＋(2.4×10^5)÷42.6－78×36.9＝2767.602817

〈操作〉 12[+]2.4[E]5[/]42.6[−]78[*]36.9[EXE]　　　2767.602817

● メモリー計算

〈 例 〉 12×45＝540

12×31＝372

75÷12＝6.25

〈操作〉	表示
A[＝]12[EXE]	—
A[✱]45[EXE]	540
A[✱]31[EXE]	372
75[／]A[EXE]	6.25

〈 例 〉 23＋9＝32

53－6＝47

－)45×2＝90

99÷3＝33

合計 22

〈操作〉	表示
M[＝]23[＋]9[EXE]	
M[＝]M[＋]53[－]6[EXE]	
M[＝]M[－]45[✱]2[EXE]	
M[＝]M[＋]99[／]3[EXE]	
M[EXE]	22

※この操作方法ですと各々の計算結果がわかりませんので、計算結果を見たいときは次の方法で行ないます。

操作	表示
23[＋]9[EXE]	32
M[＝][ANS][EXE]	
53[－]6[EXE]	47
M[＝]M[＋][ANS][EXE]	
45[✱]2[EXE]	90
M[＝]M[－][ANS][EXE]	
99[／]3[EXE]	33
M[＝]M[＋][ANS][EXE]	
M[EXE]	22

関数計算の仕方

● 三角関数(sin, cos, tan)、逆三角関数($\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$)

三角・逆三角数を使うときは、必ず角度単位の指定を行なってください。
(角度単位を変更しない場合は、新たに行なう必要はありません)。

> EXP(e)は連続計算の中ではご使用になれません。
> また、マルチステートメント文でもご使用になれません。

〈 例 〉 sin 12.3456° = 0.2138079201

〈操作〉 [MODE][4] → "DEG"
SIN 12.3456[EXE]

0.2138079201

〈 例 〉 2・sin45° × cos65.1° = 0.5954345575

〈操作〉 2[×]SIN45[×]COS65.1[EXE]

0.5954345575

〈 例 〉 $\sin^{-1}0.5 = 30°$

〈操作〉 ASN 0.5[EXE]

30

〈 例 〉 COS ($\frac{\pi}{3}$rad) = 0.5

〈操作〉 [MODE][5] → "RAD"
COS[SHIFT]([SHIFT]π[÷]3[SHIFT])[EXE]

0.5

〈 例 〉 $\cos^{-1}\frac{\sqrt{2}}{2} = 0.7853981634$rad

〈操作〉 ACS[SHIFT](SQR2[÷]2[SHIFT])[EXE]

0.7853981634

〈 例 〉 tan(−35gra) = −0.612800788

〈操作〉 [MODE][6] → "GRA"
TAN[−]35[EXE]

-0.612800788

対数関数(log,ln)、指数関数(e , x^y)

〈 例 〉 log 1.23 (= $\log_{10}$1.23) = 0.0899051114

〈操作〉 LOG 1.23[EXE]

0.0899051114

〈 例 〉 ln 90 (= $\log_e$90) = 4.49980967

〈操作〉 LN90[EXE]

4.49980967

〈 例 〉 e = 2.718281828
(この関数は指数表の数値を呼び出す命令です)

〈操作〉 EXP 1[EXE]

2.718281828

〈 例 〉 $10^{1.23}=16.98243652$

（常用対数1.23の真数を求める）

〈操作〉 10[SHIFT][x↑y]1.23[EXE]

16.98243652

〈 例 〉 $5.6^{2.3}=52.58143837$

〈操作〉 5.6[SHIFT][x↑y]2.3[EXE]

52.58143837

〈 例 〉 $123^{\frac{1}{7}}(=\sqrt[7]{123})=1.988647795$

〈操作〉 123[SHIFT][x↑y][SHIFT][(]1[÷]7[SHIFT][)][EXE]

1.988647795

※べき乗($x\uparrow y$)計算の x の値は $x>0$ です。

〈 例 〉 $\log\sin 40^\circ+\log\cos 35^\circ=-0.278567983$

その真数は……0.5265407845($\sin 40^\circ\times\cos 35^\circ$の対数計算)

〈操作〉 [MODE][4]→ "DEG"

LOG SIN 40[+]LOG COS 35[EXE]

−0.278567983

10[SHIFT][x↑y][ANS][EXE]

0.5265407845

● その他の関数（√、SGN、RAN#、RND、ABS、INT、FRAC）

〈 例 〉 $\sqrt{2}+\sqrt{5}=3.65028154$

〈操作〉 SQR 2[+]SQR 5[EXE]

3.65028154

〈 例 〉 正数であれば"1"を、負数であれば"−1"を、0であれば"0"を与える。

〈操作〉 SGN 6[EXE]

1

SGN 0[EXE]

0

SGN[−]2[EXE]

−1

〈 例 〉 乱数発生（0＜RAN#＜1の擬似乱数）

〈操作〉 RAN[SHIFT][#][EXE]

0.7903739076

〈 例 〉 12.3×4.56の答を10^{-2}の位で四捨五入する。

12.3×4.56＝56.088

〈操作〉 RND[SHIFT][(]12.3[×]4.56[SHIFT][,][−]2[SHIFT][)][EXE]

※RND(x, y)のときのyは $|y|<100$

56.1

〈 例 〉 $|-78.9\div 5.6|=14.08928571$

〈操作〉 ABS[SHIFT][(][−]78.9[÷]5.6[SHIFT][)][EXE]

14.08928571

〈 例 〉 $\frac{7800}{96}$ の整数部は……81

〈操作〉 INT SHIFT ⌐ 7800 ⁄ 96 SHIFT ⌐ EXE

81

※この関数は元の数値をこえない最大の整数を求めます。

〈 例 〉 $\frac{7800}{96}$ の小数部は……0.25

〈操作〉 FRAC SHIFT ⌐ 7800 ⁄ 96 SHIFT ⌐ EXE

0.25

● 有効桁数指定、小数以下指定

有効桁数と小数以下の指定は〝SET〟コマンドにより行ないます。

有効桁数指定……SET E n (n=0~8)

小数以下指定……SET F n (n=0~9)

指定解除…………SET N

※マニュアル計算での有効桁数指定の場合の〝SET E 0〟は8桁指定となります。

※指定を行なうと、指定桁の下1桁目を四捨五入して表示します。

なお、計算機内部やメモリー内にはもとの数値が残っています。

〈 例 〉 100÷6=16.66666666……

〈操作〉 SET E4 EXE (有効桁数4桁指定)

100 ⁄ 6 EXE

1.667E01

〈 例 〉 123÷7=17.57142857……

〈操作〉 SET F2 EXE (小数以下2桁指定)

123 ⁄ 7 EXE

17.57

〈 例 〉 1÷3=0.3333333333……

〈操作〉 SET N EXE (指定解除)

1 ⁄ 3 EXE

0.3333333333

■配　列

配列は一次元配列が使え、A(i)とかB(j)のように添字が付いた形で使われます。この配列は通常の26メモリーと増設メモリーを使いますので、次の配列の取り方に注意してください。

```
A=A(0)
B=A(1)=B(0)
C=A(2)=B(1)=C(0)
D=A(3)=B(2)=C(1)=D(0)
E=A(4)=B(3)=C(2)=D(1)=E(0)
:
Y=A(24)=B(23)=C(22)…………………………=Y(0)
Z=A(25)=B(24)= ……………………………=Y(1)=Z(0)
      A(26)=B(25)= ……………………………=Y(2)=Z(1)
増設   A(27)=B(26)= ……………………………=Y(3)=Z(2)
メモリー :
      A(93)=B(92)= ……………………………=Y(69)=Z(68)
```

このように配列を使う場合は、配列の引数により同じメモリーになることがありますので、同一プログラム中で同じメモリーを使わないようにしてください。

〈例〉

同時に使える………A、B、C、F(0)、F(9)

同時に使えない……F、G、A(5)、A(6)

なお、配列の大きさによってメモリーの増設は正確に行なってください。

プログラム計算

◈ プログラムのあらまし

プログラム計算とは①実行したい計算内容をプログラミングし、②プログラムとしたものを計算機に憶えさせ、③そのプログラムを使ってデータを入力するだけで自動的に答を求める方法です。

■ プログラミングの基本

与えられた問題をコンピュータを用いて処理するために必要な、プログラミングという概念とプログラミングの手順について検討してみます。

●プログラムとプログラミング

コンピュータ利用者が、コンピュータを使って問題を処理させる場合に、コンピュータが理解できる言葉で書かれている指令書を作成します。この指令書をプログラムと言い、指令書を作成することをプログラミングといいます。

●プログラムとは？

プログラムを作るためには、色々な文法や規則がありますが、その詳しい説明は後で述べるとして、まず、プログラムというものがどのようなもので、どのような形をしているかを簡単な基本的プログラムを例にして見てみましょう。

上記プログラムが基本的プログラムで、入力文、演算文、出力文および行番号で構成されています。すなわち、データを入力する入力文。そのデータに従い演算等の処理を行なう演算文。実行の結果を出力する出力文、そして、それぞれの行の頭には行番号が付けられています。この演算文が一つだけでなく複数回行なわれたり、判断文が加えられたりして長い複雑な演算を行なうプログラムになるわけで、基本はまったく同じです。

また、一行の中にも行番号に続いて特別な意味を持った英文字から成る単語であ

り、次に何をするか計算機に命令する「**コマンド**」とそのコマンドに必要な情報を示す文字列である「**オペランド**」が続いて書かれています。

以上がプログラムというものであり、基本的な形です。

●ステップ数の数え方

ステップ数は1機能1ステップで、プログラムに使うコマンドや関数命令も1つの命令(機能)で1ステップとなります。行番号は1～9999までの数値で、何桁であっても2ステップとなります。オペランドのような文字列やPRINT文に続くキャラクターも1文字1ステップとなります。

以上のようにプログラムステップは数えますが、この他に行区切り(EXEキーを押して記憶させたとき)として1ステップ必要とします。

〈例〉

1 INPUT A	(2, 1, 1)	5ステップ
10 B=SIN A	(2, 1, 1, 1, 1)	7ステップ
100 PRINT "B="; B	(2, 1, 4, 1, 1)	10ステップ
		計22ステップ

◈ プログラムの基本

■ 定数と変数

BASICで使用できる文字は英大文字(A,B,～Z)と数字(0,1,2……9)および若干の特殊文字(記号等)です。

●定　数

プログラム中で使われる一定の数であり、直接その数値をプログラム中に書き込むものを定数といいます。

例　$S=\pi r^2$では……S=π＊R↑2となり、2が定数です。

●変　数

プログラム中で使われる数値だが、実行中にキーボード上から入力したり、演算結果を実行中に代入するために初めからわからないような場合に使われます。

変数は英大文字1文字(A,B,C～Z)または、英大文字1文字に"**$**"をつけたもの(文字変数)であり、この範囲内で任意に選べます。

例　$S=\pi r^2$では……S＝π＊R↑2となり、Rが変数です。

例

すなわち、数学で使う代数が〝変数〞であり、定数が〝定数〞です。

また、上記の他に文字定数と文字変数があります。

文字定数は直接書き込む文字列であり〝ABC〞，〝END〞のように引用符でかこまれた文字のつづりです。文字変数は数値ではなく文字列が入る変数で、その都度文字列を与えられ内容が変わります。

なお、文字列とは〝123〞のように引用符でかこまれた文字であり数値ではありません。つまり、〝123〞というのは1と2と3が続けて書かれているだけで、〝ABC〞と同じものと考えられます。文字変数は一般変数(A,B,X,Y等)に$(ドル) マークを付けたもので、この範囲内で任意に選べます。

例　A$,B$,C$,X$,Y$

文字変数同士の比較や加算はできますが、他の演算(減・乗・除算など)はできません。

例　A$=〝123〞，B$=〝456〞ですと

C$=A$+B$

によりC$は〝123456〞となります。　(C$=B$+A$では C$=〝456123〞となります。)

この文字変数内には7文字までの文字列を入れることができます。

また、この文字変数の他に専用文字変数があります。専用文字変数は〝$〞で表わされ、文字列を30文字分入れることができます。

例　$=〝1234567890ABCDEFG〞

この専用文字変数は後で述べる文字関数(MID関数)が使えますので、他の文字変数より便利な使い方ができます。

★数値変数と文字変数は1つの英文字に対してどちらかしか使えません。

数値変数　A
文字変数　A$　＞同時には使えない

これは、数値変数も文字変数も同じメモリーを使うため、同時に使うことはできません。

■ 代入文

BASICの代入文は、

変数＝数式　　の形でできています。

BASICの代入文では、右辺の算術演算子(＋,－,＊,／)を含む式を数式といいます。

例　Y=2＊X+3では、

右辺の〝2＊X+3〟が数式です。

そして、この〝=〟は〝**等しい**〟という意味ではなく、〝**代入する**〟という意味です。

例　Y=2＊X+3 では、

左辺が変数であり右辺が数式です。

すなわち、普通の数学でいう「**左辺(Y)と右辺(2＊X+3)が等しい**」という意味ではなく、「**左辺(Y)に右辺(2＊X+3)の演算結果を入れよ**」という意味です(Y=2＊X+3はY←2＊X+3と考えるとわかりやすい)。

◆ プログラムの書き込みと実行

■ プログラムの書き込み

プログラムを計算機の記憶装置に記憶させることを、「**プログラムの書き込み**」といいます。

この操作はキーボード上からキー入力により次の手順で行ないます。

1. WRTモードの指定
2. プログラムエリアの指定
3. 行単位でのプログラムの入力(書き込み)

プログラムエリアはP0,P1,P2,………P9まで10分割でき、プログラムはこのプログラムエリアのいずれかに書き込まれます。

(1)　WRTモードの指定

プログラムの書き込みはWRTモードで行ないますので、[MODE][1]と押して〝**WRT**〟を点灯させます。

(2)　プログラムエリアの指定

プログラムエリアの指定は[SHIFT]キーに続いて[0]、[1]、～[9]の数字キーを押します。

SHIFT P0 → P0　　SHIFT P5 → P5

SHIFT P1 → P1　　SHIFT P6 → P6

SHIFT P2 → P2　　SHIFT P7 → P7

SHIFT P3 → P3　　SHIFT P8 → P8

SHIFT P4 → P4　　SHIFT P9 → P9

(3) プログラムの入力(書き込み)

プログラムの書き込みは行単位で行ない、行番号を含めて62文字以内に書き込み、最後にEXEキーを押します。

★EXEキーの役割

EXEキーは、プログラムの書き込み、データ入力、マニュアル計算の結果を求めるとき押します。

プログラムの書き込みでは、行単位のキー入力後にEXEキーを押してはじめて計算機に記憶されますので、プログラムの書き込みや書き込まれているプログラムの内容の変更、追加、削除等はすべて最後にEXEキーを押します。表示上の内容を変更してもEXEキーを押さなければ、書き込まれている内容の変更は行なわれません。

〈例〉次のプログラムをP0に書き込む。

```
10 INPUT A,B
20 V=A+B
30 W=A-B
40 PRINT V,W
50 END
```

〈操作〉

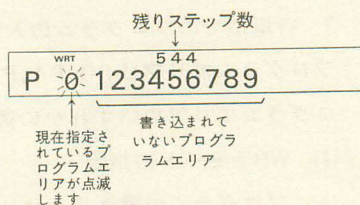
① WRTモードを指定します。

MODE 1

残りステップ数

WRT 544
P 0 123456789

現在指定されているプログラムエリアが点滅します

書き込まれていないプログラムエリア

※この表示はメモリー数や書き込まれているプログラム量により異なります。

※書き込まれているプログラムエリアはエリアNo.が表示されません。

② プログラムエリアP 0を指定します。

SHIFT P0

```
WRT         544
P  0123456789
```

③ 前に書き込んだプログラムが残っているときは、それをクリヤーします。
（書き込まれていない場合は省略可）

CLEAR EXE

```
WRT         544
P  0123456789
```

④ 10行目を書き込みます。

```
WRT         537
10 INPUT A,B
```

⑤ 20行目を書き込みます。

20␣V = A + B EXE

```
WRT         529
20 V=A+B
```

⑥ 30行目を書き込みます。

30␣W = A − B EXE

```
WRT         521
30 W=A-B
```

⑦ 40行目を書き込みます。

40␣PRINT␣V,W EXE

```
WRT         514
40 PRINT V,W
```

⑧ 50行目を書き込みます。

50␣END EXE

```
WRT         510
50 END
```

● プログラムを終了させる時には、〝END〞コマンドを書き込みます。上記のようなプログラムでは省略してもかまいませんが、GOTO文やGOSUB文を使う場合には終了する所をはっきりさせるために必ず書き込みます。

● 行番号とコマンド、コマンドとオペランドの間を1文字分あけるのは、表示上で見やすい形式にするためにあけているだけで、BASIC言語ではPRINT文等のメッセージ以外では特別の意味を持たないので省略できます。

●ここでは行番号を10刻みで入れています。1～9999の範囲内で任意に選べますが、後からの追加・挿入を考えて10刻みにした方が便利です。
なお、プログラムの実行は数値の小さい方から順に行なわれますので、実行したい順に行番号を付けてください。
●1つのプログラムエリア内のプログラムをクリヤーするには〝CLEAR″コマンドを使いましたが、P0～P9までの全部のプログラムをクリヤーするには〝CLEAR A″コマンドを使います。

■ プログラムの実行

プログラムの実行はRUNモード(MODE 0と押す……〝RUN″点灯)で行ないます。
書き込んだプログラムを実行させるには2つの方法があります。
(1)プログラムの実行方法
①プログラムエリアの指定による実行
この方法はプログラムエリアの指定と同時に実行を開始します。

〈例〉前例のプログラムをスタートさせます。
〈操作〉SHIFT P0

RUNモード(以下略)
RUN
?

※この〝?″は最初にINPUT文が書き込まれているからです。

②RUNコマンドによる実行
RUN EXE〔〝RUN″はR U Nと押してもSHIFT RUNと押しても同じです〕

?

※前例に続いて行なう場合のように、〝?″が表示され、入力待ち状態のときはACでも解除されませんので、MODE 0と操作後、②の操作を行ないます。
また、途中から実行させたい場合は〝RUN″コマンド後に行番号を置数してEXEキーを押します。
〈例〉20行目からスタートさせます。
〈操作〉RUN 20 EXE

★①の方法では実行させたいプログラムエリアに設定されていなくてもかまいませんが、②の方法では実行させたいプログラムエリアに設定されていなければなりません。(プログラムエリアが異なる場合は、そのプログラムエリアに書き込まれているプログラムが実行されます。)

(2)プログラムの実行時におけるキー入力

プログラムの実行中に行なわれるキー入力は、INPUT文によるものとKEY関数によるものがあります。KEY関数によるキー入力は1キーだけの入力ですが、キー入力のない場合でも実行は継続されます。

INPUT文によるキー入力は"?"マークが表示され、入力待ちの状態で停止しますので、データ入力後のEXEキーにより実行開始となります。

〈例〉

前例のP0に書き込まれているプログラムを実行する。

〈操作〉

● プログラムを実行させます。

SHIFT P0 　　 | ? |

● このプログラムでは2つの変数を入力しますので、まず変数Aの値を入力します。

47 EXE 　　 | ? |

● 次に変数Bの値を入力します。

69 EXE 　　 | 116 |

EXE 　　 | −22 |

このように、INPUT文による実行中のキー入力は、

データ EXE

というように、データを入力させます。

また、INPUT文により入力待ちの状態のときに、マニュアル計算等の他の操作が行なえます。

なお、入力待ち状態のときにプログラムの実行を中止したいときはMODE ⊘と押せば中止となります。

◆ プログラムの編集

- プログラムの編集とは、プログラムを論理上正しく、実行可能なものとするために、行単位で変更・追加・削除をしたり、行番号のつけ直しをすることです。
- プログラムの編集はLISTコマンドにより行ごとに呼び出して行ないます。
- LISTコマンドはRUNモードとWRTモードの両方で使えますが、RUNモードで使う場合はプログラム内容の表示となり、WRTモードで使う場合はプログラムの編集となります。

(1)RUNモードにおけるプログラムリストの表示

〈操作〉 LIST EXE
(LISTは L I S T または SHIFT LIST)

10 INPUT A,B ………約2秒間表示されます。(以下同じ)
20 V=A+B
30 W=A−B
40 PRINT V,W
50 END
READY P0

なお、最初からのリストが必要でない場合は行番号を指定します。

30行目以降をリストするには
〈操作〉 LIST 30 EXE

30 W=A−B
40 PRINT V,W
50 END
READY P0

※LISTコマンド実行中は順番に最後まで表示されますので、停止させたい場合は STOP キーを押します。
また、停止したLISTコマンドを続行させるときは EXE キーを押します。

(2)WRTモードにおけるプログラムの変更・追加・削除

MODE 1 と押して〝WRT〞モードを指定します。

①変　更

LISTコマンドにより指定された行番号から EXE キーを押すごとに一行づつ表示されます。

また、指定された行番号が省略された場合は最初の行より順に表示されます。

a 一部変更

〈 例 〉 前例の20行目の"+"を"*"に変更する。

〈操作〉

● プログラムエリアがP0に指定していない場合は、P0に指定します。

SHIFT P0

```
         510
P  ☀123456789
```

└点滅は書き込まれていて、現在指定されているプログラムエリアです。

● LISTコマンドにより20行目を呼び出します。

LIST 20 EXE

```
         510
20 V=A+B_
```

● カーソルを動かし、変更したい箇所"+"に合わせます。

⇦ ⇦

```
         510
20 V=A±B
```

※カーソル移動キー(⇦⇨)は約1秒以上押し続けますと、早送りができます。

● 変更箇所を直します。

＊ EXE

```
         510
30 W=A-B_
```

※EXEキーは必ず押してください。押さない場合は表示上のみの変更となり、書き込まれているプログラム内容は変更されません。

● このままですと30行目の変更となりますので、クリヤーして変更を終了させます。

AC

```
         510
_
```

※変更の必要ない行での他のキー操作は、その行への書き込みとなりますのでEXE、AC以外は押さないでください。

● 変更を確かめる意味でリストを見ます。

MODE ⊘

LIST EXE

```
READY P0
10 INPUT A,B
20 V=A*B
30 W=A-B
40 PRINT V,W
50 END
READY P0
```

b 一行全部の変更

変更する行番号と同じ行番号を入力します。(これにより、前に入力されていた行番号がクリヤーされます。)

〈例〉30行目の〝W=A−B″を〝W=V/2″に変更する。

〈操作〉 MODE 1

P 123456789 (510)

● 新しく30行目を書き込みます。

30␣W=V/2 EXE

30 W=V/2 (510)

● プログラムリストを確認します。

MODE 0

LIST EXE

```
READY P0
10 INPUT A,B
20 V=A*B
30 W=V/2
40 PRINT V,W
50 END
READY P0
```

②追 加

行単位の追加は新たに追加したい行と行の間の番号をつけて書き込みます。

〈例〉前例のプログラムの30行目と40行目の間に〝U=V*2″を追加し、40行目を〝PRINT V,W,U″に変更する。

〈操作〉 MODE 1

P 123456789 (510)

● 30行目と40行目の間に入れるため、行番号35で入力する。

35␣U=V*2 EXE

35 U=V*2 (502)

※30行目と40行目の間であれば31～39の範囲内で任意に選べます。

● 40行目を変更するためにLIST 文で呼び出し、〝,U″を追加する。

LIST 40 EXE

⇨ SHIFT ,U EXE

AC

```
40 PRINT V,W   (502)
50 END         (500)
_              (500)
```

●追加されたプログラムを確認するためにリストを見る。

MODE 0

LIST EXE

```
READY P0
10 INPUT A,B
20 V=A*B
30 W=V/2
35 U=V*2
40 PRINT V,W
PRINT V,W,U
50 END
READY P0
```

③削　　除

a 一部削除

〈例〉前例の40行目から"V,"を削除する。

MODE 1

```
P  123456789
```

●一部変更の方法と同様に、40行目をLIST文で呼び出す。

LIST 40 EXE

```
40 PRINT V,W
```

●カーソルを動かし、削除したい箇所の"V"に合わせる。

⇦ ⇦

```
40 PRINT V,W
```

●DEL キーにより"V,"を削除する。

DEL DEL

EXE

```
40 PRINT W,U
50 END
```

※このEXEキーを押さなければ、プログラム内容は変わりません。

AC

```
_
```

※このACは、50行目の変更状態を解除するために必ず押します。

●削除された箇所を確認するために、
リストを見る。

MODE 0

LIST EXE

```
READY P0
10 INPUT A,B
20 V=A*B
30 W=V/2
35 U=V*2
40 PRINT W,U
50 END
READY P0
```

b 一行全部の削除

削除したい行番号と同じ行番号だけを入力すれば、その行が全部クリヤーされます。

〈例〉30行目を削除する。

〈操作〉MODE 1

```
         502
P ※123456789
```

●削除したい行番号"30"を入力する。

30 EXE

```
         510
_
```

●削除された箇所を確認する。

MODE 0

LIST EXE

```
READY P0
10 INPUT A,B
20 V=A*B
35 U=V*2
40 PRINT W,U
50 END
READY P0
```

④行番号のつけ直し

〈例〉以下のプログラムがP2に書き込まれています。

```
10 INPUT N
20 M=N*N
30 L=SQR N
40 PRINT M,L
50 END
```

20行目を30行目と40行目の間に移動させる。

〈操作〉 MODE 1

```
P _1 3456789
```

● LIST コマンドにより20行目を呼び出します。

LIST 20 EXE

```
20 M=N*N_
```

● カーソルを動かし、行番号"20"の"2"に合わせる。

⇦⇦⇦⇦⇦⇦⇦⇦

```
20 M=N*N
```

● 20を35に変更して入力します。

35 EXE

```
30 L=SQR N _
```

● 変更終了のため AC を押して変更命令を解除します。

AC

```
_
```

● プログラム内容がどのように変更されたかを、リストします。

MODE 0

LIST EXE

```
READY P2
10 INPUT N
20 M=N*N
30 L=SQR N
35 M=N*N
40 PRINT M,L
50 END
READY P2
```

● このままでは20行目にあった内容が30行目と40行目の間に移動しましたが、まだ20行目にも残っていますので、不必要な行を削除します。

MODE 1

20 EXE

471
P _1_3456789
479
_

● これで行番号のつけ直しは終了ですので、リストで確認します。

MODE 0

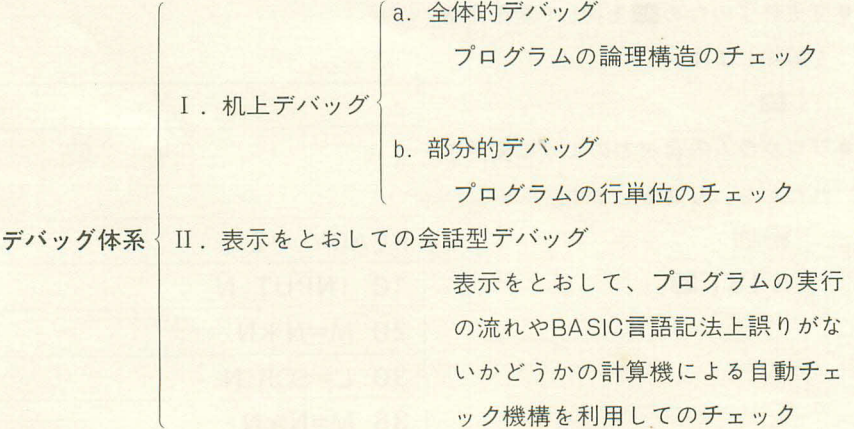
LIST EXE

READY P2
10 INPUT N
30 L=SQR N
35 M=N*N
40 PRINT M,L
50 END
READY P2

◆ プログラムのデバッグ

(1)プログラムのデバッグ体系

本機のデバッグ体系は、机上デバッグと表示をとおしての会話型デバッグに大別されます。

デバッグ体系
- Ⅰ．机上デバッグ
  - a. 全体的デバッグ
    プログラムの論理構造のチェック
  - b. 部分的デバッグ
    プログラムの行単位のチェック
- Ⅱ．表示をとおしての会話型デバッグ
  表示をとおして、プログラムの実行の流れやBASIC言語記法上誤りがないかどうかの計算機による自動チェック機構を利用してのチェック

机上デバッグはプログラミングのときに行なわれますので、ここでは表示をとおしての会話型デバッグについて説明します。

⑵会話型デバッグ

プログラムの実行段階で発生するエラーは表示窓にエラーメッセージが表示されます。このエラーは行単位で示され、BASIC言語上どのようなエラーであるかが指示されますので、表示に表示されたエラーメッセージに基づき、表示をとおして会話しながらマニュアルでデバッグ処理を行ないます。

なお、エラーメッセージの意味は62ページの**「エラーメッセージ一覧表」**をご覧ください。

〈例〉

```
10 INPUT X
20 IF X≦0;PRINT "X≦0":GOTO 10
30 Y=X↑2+3*X+15
40 PRINT Y
50 END
```

このプログラムの20行目はべき乗計算の入力範囲の判断で、x≦0のときは10行目の入力文に戻ります。

このプログラムの30行目を

```
30 Y=X↑2+3X+15
```

と誤って書き込んでしまった場合。

〈操作〉

● このプログラムを実行させると、10行目のINPUT文により"?"が表示されます。

操作	表示
RUN EXE	?

● このときにたとえば"45"を入力します。

操作	表示
45 EXE	ERR2 P0-30

● このエラーメッセージは**「30行目で構文エラーが発生しました」**という意味ですので、プログラム内容を確認します。

操作	表示
AC MODE 1	P _123456789
LIST 30 EXE	30 Y=X↑2+3X±

●30行目の〝3″と〝X″との間に〝*″が抜けていますので、プログラムの編集の方法に従い直します。

キー操作	表示
⇦ SHIFT INS	30 Y=X↑2+3_X
* EXE	40 PRINT Y

(3)**プログラムを実行しながらのデバッグ**

会話型デバッグは、エラーメッセージにより計算機からの情報を得て行なうデバッグですが、エラーメッセージは表示されないが計算結果が思い通りにいかない場合には、プログラムを繰り返し実行しながら途中までの計算結果を確認していくデバッグを行ないます。この方法は、〝**STOP**″コマンドによりプログラムの進行を停止させて行なう方法と、トレースモードにより一行ごとに実行させてデバッグを行なう方法があります。

●STOPコマンドによるデバッグ

〈例〉次のプログラムが書き込まれています。

```
10 Y=0
20 INPUT N,X
30 FOR I=1 TO N
40 Y=Y+X*X
50 NEXT I
60 PRINT Y
70 END
```

このFOR·NEXTループ中のYの値を見るために、STOP文により一回ごとのループの結果を見る。

〈操作〉

●STOP文を入れる箇所は計算式の直後がよいので、40行目と50行目の間にSTOP文を書き込む。

MODE 1

45 STOP EXE

●これにより、40行目の計算が終了後プログラムの進行が止まりますので、チェックすることができます。

MODE 0 RUN EXE
4 EXE
87 EXE

●この停止しているときのYの値は？

Y EXE

●再びプログラムをスタートさせますと、次のSTOP文で停止しますのでまたYの値を求めます。

キー操作	表示
EXE	_ (STOP)
Y EXE	15138 (STOP)

●この操作の繰り返しで、計算過程を見ることができます。

この例は簡単なプログラムを使っていますが、実際に複雑なプログラムを組み上げていく場合には机上デバッグでは計算過程のチェックが大変なため、このようなSTOP文による変数のチェックを行なえば、プログラムミスを見つけだし、直すことができます。

●トレースモードによるデバッグ

トレースモード（MODE 2と押す）でプログラムの実行を行ないますと、一行ごとに実行して止まりますので、実行デバッグが簡単に行なえます。

前記STOPコマンドによるデバッグの例題をトレースモードで行なってみます。

〈操作〉

操作	キー	表示
RUNモード指定	MODE 0	READY P0
トレースモード指定	MODE 2	READY P0 (TR)
	RUN EXE	P0−10 (TR, STOP)
	EXE	? (TR)
プログラムの進行を見る	STOP	? (TR, STOP)
	STOP	P0−20 (TR, STOP) ………〝TR〞〝STOP〞は以下略
プログラムを続ける	EXE	?

キー操作	表示
4 EXE	?
87 EXE	_
STOP	P0−20
EXE	P0−30
EXE	P0−40
Yの値は……………………Y EXE	7569
EXE	P0−45
⋮	⋮

以下繰り返し

このトレースモードによるデバッグは、全体の流れを見るのに最適で、どの箇所でミスがおきているかを見るのに便利です。

◈ プログラムコマンド

■ 入力命令

● INPUT文

入力コマンドは、プログラム計算実行中にデータを入力するためのコマンドです。INPUT文はプログラム実行中に、変数にデータをキー入力するためのコマンドで"?"を表示して停止します。

書　式：INPUT 〔"**文字列**",〕変数〔,"**文字列**",変数〕　（〔 〕内は省略可）

"**文字列**"は省略できますが、書き込まれた場合は"?"の前に""(引用符)でかこまれた文字が表示されますので、入力時のメッセージとすることができます。

続く変数は、数値変数(A,B等)、文字変数(X$,Y$等)、専用文字変数($)で、,により続けて書くことができます。

〈例〉

プログラム	表示
INPUT A	?
INPUT "DATA=",A	DATA=?

INPUT文により"?"を表示して入力待ち状態のときは、データを入力してEXEキーを押せば、次の処理に進行します。なお、入力待ち状態のときはACキーを押しても入力待ち状態は解除されませんので、プログラムを途中で中止したいときはMODE 0と操作します。

★INPUT文により入力できるデータは、原則として数値変数には数値もしくは数式の結果(答)が、文字変数には文字列が入ります。

INPUT A の場合

数値……123EXE → A=123

数式の結果……14×25EXE → A=350

INPUT B$の場合

文字列……ABCEXE → B$=ABC

789EXE B$=789

なお、数値変数への入力に他の数値変数を使うことができます。

INPUT A の場合（X＝987654とする）

変数……XEXE ——→ A=X

=987654

● KEY関数

この関数は、プログラム実行中に押されている1キー分だけを1文字の文字として文字変数に読み込みます。この関数はINPUT文と異なり、停止して入力待ち状態("?"表示)にはならず、キー入力がない場合にもプログラムは順次進行します。

書　式：　文字変数＝KEY

文字変数はA$、$等を使います。

〈 例 〉

```
10 A$=KEY
20 IF A$= "A" THEN 100
30 IF A$= "B" THEN 200
40 IF A$= "C" THEN 300
50 GOTO 10
   :
```

このプログラムはKEY関数によるデータの入力と振り分けの部分だけですが、10行目のKEY関数により読み込まれた文字データを、次のIF文によりキー入力されたかどうかを判断します。これは、KEY関数はEXEキーを押さなくても第1キー入力だけを読み込む働きがありますが、INPUT文のように停止はしませんので、次

のIF文との組み合わせにより入力待ちの状態にします。

20～40行目のIF文は判断のためのコマンドで、KEY関数により入力された文字変数により振り分けます。なお、IF文についての詳細は48ページをご覧ください。

■ 出力命令

●PRINT文

PRINT文は、計算結果やデータを表示させる命令文で、コマンドに続く文字列や変数の内容、計算結果が表示されます。

書　式：PRINT 〔出力制御関数（CSR）〕〔{数　式 / 文字式}〕〔{, / ;}……〕

{ }内はいずれか1つ　〔 〕内は省略可

PRINT文に続く出力制御関数はCSR関数で後に続くデータを表示する位置を指定します。(47ページ参照)

数式は変数や計算式を書きます。変数の場合は変数の内容が、計算式の場合は計算の結果が表示されます。

〈例〉

PRINT A　(A＝12345とします)	12345
PRINT 789	789
PRINT A*2 (A＝147とします)	294
PRINT B$ (B$＝"PB-100"とします)	PB-100

文字式の場合は" "(引用符)でかこまれた文字がそのまま表示されます。

〈例〉

PRINT "ABC"	ABC
PRINT "XYZ" + "123"	XYZ123

この数式や文字式は";"または","によりいくつも続けて書くことができますが、一行に書ける文字数は行番号を含めて62文字以内で、引用符間の文字列は30文字以内となります。

";"と","の違いは、後に続く数式や文字式を前の式に続けて表示させるか、一度表示を消してから表示させるかです。

データの後に";"が続かない場合はデータを表示後"STOP"を表示して停止し

ますので、次のデータを表示させたいときやプログラムを進行させたいときはEXEキーを押します。

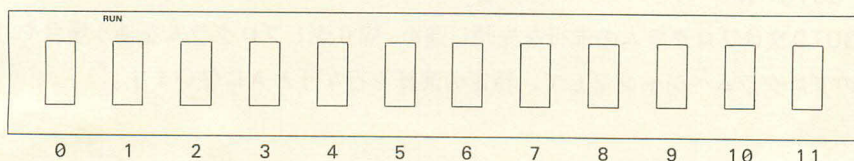
●CSR関数

CSR関数は出力制御関数で、データを表示させる位置を指定する関数です。

書　式： PRINT CSR 数式※ 〔{ , ／ ; }{ 数　式 ／ 文字式 }……〕　〔 〕内は省略可　{ }内はいずれか一つ

※数式の値は小数以下切り捨てで0～11です。

この数式の値により表示上の左から何桁目にデータを出力するかを指定します。なお、表示上の桁数の数え方は次のようになります。

〈例〉

プログラム		表示
PRINT A	(A=12345とします)	12345
PRINT CSR 1;A		12345
PRINT CSR 5;A		12345
PRINT B$	(B$=ABCDEとします)	ABCDE
PRINT CSR 2;B$		ABCDE
PRINT CSR 10;B$		AB
		ABCDE

※CSR関数の後に続く〝;〟を〝,〟とした場合は、一度表示をクリヤーにしてから次のEXEで左側から表示されます。

■ジャンプ命令

●GOTO文

GOTO文は無条件ジャンプとも呼ばれるように、指定した箇所(行番号)に無条件にプログラムを進める命令です。

書　式： GOTO { 数式………行番号(1～9999の範囲) ／ #数式………プログラムエリアNo.(0～9の範囲) }

GOTO文の後に直接数式が続く場合は行番号へのジャンプとなり、"#"と続く場合はプログラムエリアへのジャンプとなります。

数式とは、数値または変数、計算式です。

〈例〉

GOTO 10 ………10行目へジャンプ

GOTO N ………変数Nの値の行番号へジャンプ

GOTO A*100………A*100の答の行番号へジャンプ

GOTO #2 ………P2のプログラムエリアへジャンプ

GOTO #X ………変数Xの値のプログラムエリヤへジャンプ

GOTO #P+1 ………P+1の答のプログラムエリアへジャンプ

GOTO文はプログラムの進行を先頭に進め、繰り返しプログラムを使う場合や、別のプログラムへジャンプして、特定の演算を行なうときに使います。

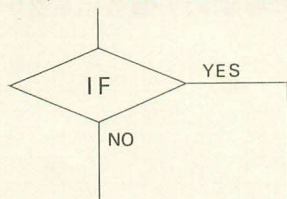
■断判命令

●IF文

IF文はその性質上条件ジャンプとも呼ばれ、ある種の条件を満たしたときにだけある操作を行なったり、指定箇所へジャンプする命令です。

書　式：IF 比較式 { THEN 行番号または#n (n=0〜9) / ; コマンドまたは代入文 }

"IF"の後の比較式は等号または不等号の右辺と左辺の比較により、YES ならば"THEN"または";"以後に続き、NOならば次の行に進みます。

これをフローチャートで描くと次のようになります。

この意味は、IF文が成立すれば"YES"(正)の方へ行き、成立しなければ"NO"(否)の方へ行きます。

つまりIF文とは分岐を示すものであり、そのために判断を行ないその結果によって次の操作を選択させます。

このIF文は、データ数がわからない時のループ(繰り返し)を終了させるためや、演算結果等により次の操作が変わるような時に使われます。

この比較には、定数・変数・数式・文字定数・文字変数が使えます。

A>10	変数と定数(Aが10より大きければYES)
X≧Y	変数と変数(XがYよりも大きいか等しければYES)
N=L+3	変数と数式(NがLと3を加算したものに等しければYES)
A$="XYZ"	文字変数と文字定数(A$内の文字列が"XYZ"と等しければYES)
P$=Q$	文字変数と文字変数 (P$内の文字列とQ$内の文字列が等しければYES)

※変数と文字変数の比較はできません。

※文字列の比較はアスキーコードに準じます。

"THEN"または";"は、次に続くものにより使いわけます。

THEN 150	(行番号)	;PRINT A
THEN #9	(プログラムエリア)	;Z=X+Y

■ ループ命令

●FOR·NEXT文

FOR·NEXT文とは繰り返して同じ(もしくは同じような)操作を行ないたい時に、その繰り返し(ループ)回数がわかっている場合に使われます。

書 式: FOR 変数=$\underline{n}$ TO $\underline{m}$ [STEP $\underline{\ell}$]

初期値 終値 増分

∫

NEXT 変数

〔 〕内は省略可

n, m, ℓは数式

つまり、変数がnからmまで刻み幅ℓで変化していく間に"FOR"と"NEXT"の間の命令を繰り返して実行せよという命令で、mまで実行すれば"NEXT"の次の命令に進みます。

〈例〉

変数Iが1から10まで2づつ増えていく場合

```
FOR I=1  TO 10  STEP 2
    ∫
NEXT I
```

変数Aが50から1まで0.5づつ減る場合

```
FOR A=50  TO 1  STEP−0.5
    ∫
NEXT A
```

変数PがQからRまで1づつ増える場合

```
FOR P =Q  TO  R
    ∫
NEXT P
```

※1づつ増える場合は〝**STEP**〟が省略できます。

★ネスティング

FOR·NEXTループは4重まで重ねて使えます。 この重ねて使うことをネスティングと言います。

これは4重のネスティングで、FOR·NEXT文の間に別のFOR·NEXT文が入り、さらにもう一つのFOR·NEXT文が入り……という形になります。

このようにいくつも重ねる場合は、FOR文に対応するNEXT 文とその変数に注意しなければなりません。

```
  ┌FOR  I=1   TO 5   STEP 1
 ┌┼FOR  J =2   TO 20   STEP 2
× ││    ∫
 │└NEXT  I
 └─NEXT  J
```

このようなFOR·NEXTループは組めません。

なお、FOR·NEXTループからの抜け出しはできますが、FOR·NEXTループへの飛び込みはできません。

■ サブルーチン命令

●GOSUB文

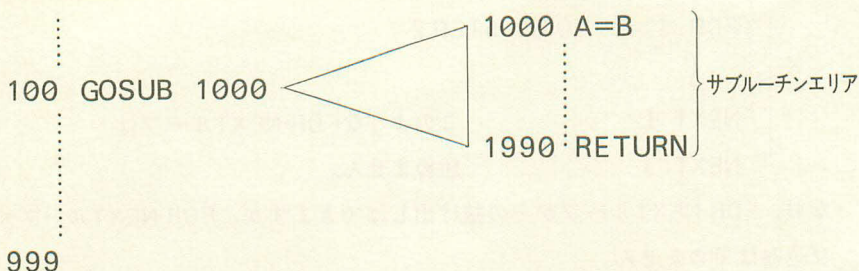
サブルーチンとは、副プログラムとも呼ばれ、今迄出てきたプログラム（メインルーチンとも呼ばれる）とは異なり、メインルーチンの中から呼ばれる別のプログラムのことです。

このサブルーチンを呼び出す命令がGOSUB文で、この命令によりメインルーチンからサブルーチンへジャンプし、サブルーチン内のプログラムを実行後、サブルーチン内のRETURN文によりメインルーチンの元の場所に戻ります。

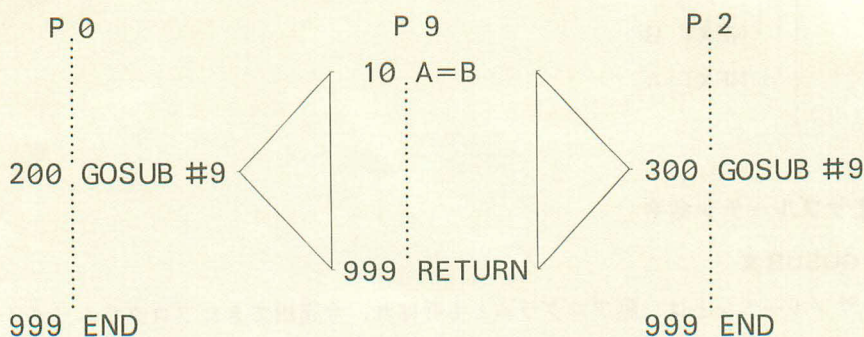
書　式：　GOSUB ｛数式 ／ ＃数式｝ ……………サブルーチン呼び出し(ジャンプ)命令

　　　　　RETURN ……………………メインルーチンに戻る命令

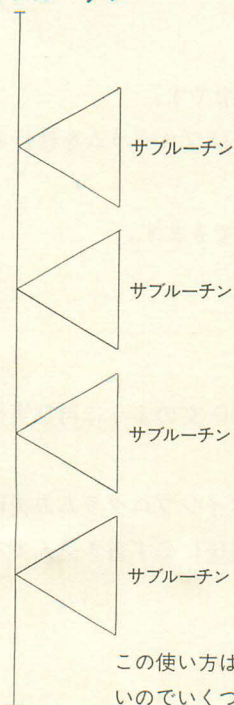
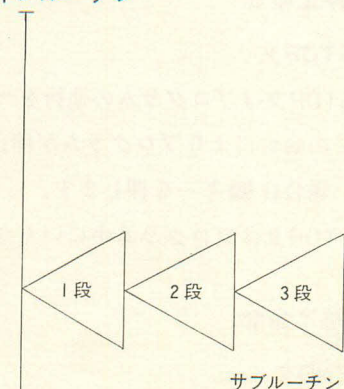
GOSUB文に続く数式はサブルーチンエリアの先頭の行番号を示します。RETURN文はサブルーチンエリアの最後に必ず入れなければならず、入れない場合はメインルーチンに戻ってきません。

〈例〉

GOSUB文の後の数式は数値変数でも計算式でもよく、変数や数式の場合は変数内の数値や数式の結果(答)により、呼び出すサブルーチンが異なります。

数式の前に〝#〟がついた場合は、別のプログラムエリア(P0～P9)をサブルーチンとして使います。この使い方は、別々のプログラムでも同じサブルーチンを使うことができますので、とても便利です。

〈例〉

● **ネスティング**

GOSUB文にもFOR·NEXT文と同様なネスティングがあり、サブルーチンを続けて呼び出す回数が決っています。

このネスティングは8段まで使えますので、サブルーチンの中からさらにサブルーチンを呼び出すことができます。

サブルーチンはメインルーチンの共通部をサブルーチンとして組み、ステップ数を省略させるために使ったり、複雑なプログラムを組み上げる時に部分部分を別々にサブルーチンとして組むと便利です。

■ マルチステートメント

マルチステートメントとは2つ以上のコマンドを“:”(コロン)でつないで、いくつもの行にわたるプログラムを一行にします。

〈例〉

```
10 A=2
20 B=10    } 10 A=2:B=10:C=50
30 C=50

10 PRINT "NO." ;N;
20 INPUT A        } 10 PRINT "NO." ;N;:INPUT A
```

〔注意〕

VAC(メモリクリヤー)コマンドはマルチステートメントで使えません。

■ 停止命令

● **STOP文**

STOP文はプログラムの進行を一時的に停止させる命令です。

この命令によりプログラムが停止しているときに、再びプログラムを進行させたい場合は EXE キーを押します。

STOP文はプログラム中にいくつでも書き込むことができます。

■ 終了命令

● **END文**

END文はプログラムの実行を終了させる命令で、STOP 文のように再び実行を継続させることはできません。

このEND文はプログラムの最後に書き込みますが、メインプログラムの後にサブルーチンが続くような場合には、メインルーチンの最後に必ず書き込んでください。

■ 実行命令

● **RUNコマンド**

RUNコマンドはプログラムを実行させるためのコマンドで、プログラム中に書き込んで使うことはできません。

書　式：　RUN 〔行番号〕　　〔 〕内は省略可

行番号が続く場合はその行番号からプログラムがスタートし、省略された場合は先頭の行からスタートします。

〈 例 〉

RUN　　EXE………先頭からスタート

RUN 20 EXE………20行目からスタート

RUN 55 EXE………55行目からスタート

指定された行番号が無い場合には、その行番号に近い次の行からスタートします。

■ リスト命令

●LISTコマンド

LISTコマンドはプログラム内容を表示させる命令で、〝RUN″モードと〝WRT″モードで使うことができます。

書　式：　LIST　〔行番号〕　　　〔 〕内は省略可

　　　　　LIST　A

行番号が続く場合は、現在指定されているプログラムエリアのプログラムを、指定された行番号から順に表示します。行番号が省略された場合は先頭の行から表示します。

〝LIST A″となった場合は、全プログラムエリア内のプログラムを表示する命令となり、P0から順にP9までのプログラム内容を表示します。

なお、このコマンドはプログラム中に書き込んで使うことはできません。

〝RUN″モードで行なった場合は指定行から順にプログラム内容を表示していきますが、〝WRT″モードで行なった場合はEXEキーを押すごとに一行づつ表示します。

〈例〉

〝RUN″モード

LIST EXE

10 A=0
20 INPUT B
30 A=A+B
40 GOTO 20

〝WRT″モード

キー	表示
LIST 20 EXE	20 INPUT B
EXE	30 A=A+B
EXE	40 GOTO 20

なお、〝WRT″モードで行なった場合はプログラムの編集(34ページ参照)となりますので注意してください。

■ モード指定

●MODEコマンド

MODEコマンドは角度単位やプリンタ出力の状態をプログラム中で指定するコマンドです。

書　式：　MODE n　(n=4～8)

MODE 4　〝DEG″指定 ⎫
MODE 5　〝RAD″指定 ⎬ 角度単位の指定
MODE 6　〝GRA″指定 ⎭

MODE 7　プリントモード指定

MODE 8　プリントモード解除

このMODEコマンドは、マニュアル操作で行なう[MODE]キーによる指定と同じで、プログラム中に書き込んで行なう方法です。

■ 出力フォーマット

●SETコマンド

SETコマンドは、表示するときの出力形式を指定する命令で、有効桁数と小数点以下を指定します。

書　式：　SET　E　n………有効桁数指定

SET　F　n………小数以下指定

SET　N　　………指定解除

(n=0~9)

※プログラム中での有効桁数指定の場合の"SET E0"は10桁指定となります。

この命令はマニュアルで行なう操作をプログラム中に書き込んで使う方法で、出力形式を指定します。表示内容は24ページを参照してください。

■ 文字関数

●LEN

LEN関数は文字変数内の文字数を数える関数で、文字変数の大きさを知ることができます。

書　式：　LEN（文字変数）

〈例〉

A$= "ABCDE" であれば

LEN(A$)=5　となります

●MID

MID関数は専用文字変数($)のみに使え、$変数内の文字列の中からいくつかを取り出します。

書　式：　MID（m〔,n〕）　m, nは数式で1以上30以内です〔 〕内は省略可

これは専用文字変数内の文字列の第m番目からn文字を取り出します。

数式mは記憶されている文字数を越えてはならず、また$m+n$が記憶されている文字数＋1を越えてはなりません。

なお、数式nが省略された場合は第m文字以降すべてを取り出します。

〈例〉

$＝"PB－100"であれば

MID (2,3)＝"B－1"

MID (4)＝"100"となります。

● **VAL**

VAL関数は文字変数内の数字を数値に変換します。

書　式：　VAL (文字変数)

この関数は文字変数内の数字を数値に変換しますので、文字変数内が数字でない場合("ABC"等)はエラーとなります。

〈例〉

Z$＝"78963"であれば

VAL (Z$)＝78963となります

〔注意〕

この関数をプログラム中で使ったときに、変数内のデータが数字以外でエラーとなった場合は"ERR2"と表示され、プログラムエリアと行番号は表示されません。

■ メモリークリヤー

● **VAC**

VACコマンドは変数内のデータをすべてクリヤーし、数値変数は"0"に、文字変数はNull (何もない)にします。

このコマンドはマニュアルでもプログラム中に書き込んでも使えますので、必ずデータをクリヤーにしてからプログラムを実行したい場合はプログラムの先頭に入れ、また、クリヤーする必要がないときや、状況に応じて必要なときはマニュアルで実行させます。

〈例〉プログラムに書き込む

10　VAC

マニュアルで実行させる

VAC EXE

〔注意〕

このVACコマンドはマルチステートメント(53ページ参照)では使えませんので、必ず一行にVACコマンドだけを書き込んでください。

■ プログラムクリヤー

●CLEARコマンド

CLEARコマンドは書き込まれているプログラムをクリヤーする命令で、"WRT"モードでマニュアルにより実行します。

書　式：　CLEAR

　　　　　CLEAR　A

"CLEAR"コマンドは現在指定されているプログラムエリア(P0、P1……)1つだけのプログラムをクリヤーします。

"CLEAR　A"コマンドは、P0～P9までの全プログラムエリアのプログラムをクリヤーします。

〈例〉

MODE 1　CLEAR　EXE………単一プログラムのクリヤー

MODE 1　CLEAR　A　EXE………全プログラムのクリヤー

■ オプション仕様

カセットテープレコーダー

本機でカセットテープにプログラムやデータを記録するには、カセットインタフェイス〈FA－3〉を使用して、一般の音楽用テープレコーダに録音します。また、テープレコーダーはリモート端子付きであれば、FA－3を通して本体からのリモートができますので、リモート端子付きの方が便利です。

なお、テープレコーダーへの端続の仕方および詳しい取り扱い方法はFA－3の取扱説明書をご覧ください。

● プログラムの記録

書　式：SAVE 〔"ファイル名"〕　　〔 〕内は省略可

ファイル名は8文字以内の英文字・数字・記号を""(引用符)でかこって使います。(以下同)

〈例〉

"ABC"

"NO.1"

このコマンドは、テープレコーダーを「録音」でスタートさせ、

SAVE 〔"ファイル名"〕 EXE

と操作します。

SAVEコマンドはマニュアルでのみ使えます。

● プログラムの呼び戻し

書　式：LOAD 〔"ファイル名"〕　　〔 〕内は省略可

このコマンドは、テープレコーダーを「再生」でスタートさせ

LOAD 〔"ファイル名"〕 EXE

と操作します。

プログラムをLOAD中の表示

このコマンドは呼び戻す前のプログラムエリアにすでにプログラムが書き込まれている場合は、新しいプログラムの先頭の行番号以外のプログラムが消えて、新たにLOADされます。

● 全プログラムの記録

書　式：SAVE A 〔"ファイル名"〕　　〔 〕内は省略可

このコマンドは、P0〜P9までの全プログラムエリア内に書き込まれているプログラムを同時にテープに記録します。

操作方法はSAVEコマンドと同じで、テープレコーダーを「録音」でスタートさせ、

SAVE A 〔"ファイル名"〕 EXE

と操作します。

● **全プログラムの呼び戻し**

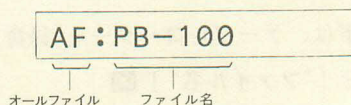
書　式：LOAD A〔"**ファイル名**"〕　　〔 〕内は省略可

このコマンドはSAVE Aコマンドにより記録された全プログラムエリアのプログラムを同時に呼び戻します。

操作方法はLOADコマンドと同じで、テープレコーダーを「**再生**」でスタートさせ、

LOAD A〔"**ファイル名**"〕 EXE

と操作します。

プログラムをLOAD中の表示

このコマンドは呼び戻す前のプログラムエリアにすでに書き込まれていても、以前のプログラムをクリヤーしてから新たにプログラムを呼び戻します。

なお、SAVE Aコマンドも LOAD Aコマンドもマニュアルでのみ使えます。

● **データの記録**

書　式：PUT〔"**ファイル名**"〕変数1〔,変数2〕　〔 〕内は省略可

テープに記録されるデータは変数1～変数2までの変数内のデータです。

〈例〉

PUT "**PB**" A　　………変数Aのデータ

PUT "**1-2**" A,Z ………変数A～Zのデータ

PUT "**DT**" $,A,Z(10)………文字変数$とA～Z(10)のデータ

専用文字変数$内のデータを記録する場合は$を最初に書きます。

このコマンドはマニュアルでもプログラム中に組み込んでも使えます。

マニュアルでは、テープレコーダーを「**録音**」でスタートさせ、

PUT〔"**ファイル名**"〕変数1〔,変数2〕 EXE

と操作します。

プログラム中に組み込んで行なう場合は行番号とともにPUTコマンドを書き込み、書き込まれたプログラムをスタートさせます。

● **データの呼び戻し**

書　式：GET〔"**ファイル名**"〕変数1〔,変数2〕　〔 〕内は省略可

このコマンドもマニュアルでもプログラム中に書き込んでも使えます。

マニュアルではテープレコーダーを「再生」でスタートさせ、

GET〔"**ファイル名**"〕変数1〔,変数2〕 EXE と操作します。

プログラム中では行番号を付けて書き込み、プログラムをスタートさせます。

●テープに記録されたファイルのチェック

テープ上に正しくプログラムやデータが記録されたかどうかをチェックするにはVERコマンドを使います。

書 式：VER〔"**ファイル名**"〕 〔 〕内は省略可

操作手順はプログラムのLOADと同様に行ないます。

プリンタ

本機には専用のミニプリンタ<FP-12>を接続することができます。

このプリンタを接続することによりプログラムリストやデータリストがとれ、また実行時の演算結果の出力もプリントすることができます。

なお、プリンタへの接続および操作方法はFP-12の取扱説明書をご覧ください。

プログラムリストをプリントしたい場合はMODE 7と押して"**PRT**"モードを指定して行ないます。

プログラムリスト

MODE 0 MODE 7

LIST EXE または LIST A EXE

MODE 8 (プリントモード解除)

プリントが終りましたら必ずMODE 8と押してプリントモードを解除してください。

また、演算結果や操作内容をプリントしたい場合は、プログラム中に"**MODE 7**""**MODE 8**"を書き込むことにより自動的にプリントすることができます。

〈例〉

```
 ⌠
100 MODE 7
110 PRINT A
120 MODE 8
```

プログラム中に"**MODE 7**"を書き込んだ場合は、必ずプログラム終了前に"**MODE 8**"を書き込んで、"**PRT**"モードを解除してください。

エラーメッセージ一覧表

エラーコード	意味	エラー原因	対策
1	メモリーオーバーまたはシステムスタックオーバー	●ステップ数不足でプログラムが書き込めない。またはメモリーが増設できない。 ●計算式が複雑すぎてスタックオーバーしている。	●不要なプログラムをクリヤーするか、メモリー数を減らす。 ●数式を分割して簡単にする。
2	構文エラー	●プログラム等に書式上の誤りがある。 ●代入文等で左辺の型式と右辺の型式が異なる。	●入力したプログラム等の誤りを修正する。
3	数学的エラー	●数式の演算結果が10^{100}以上の場合。 ●数値関数の入力範囲外の場合。 ●結果が不定または不能となる場合。	●演算式またはデータを修正する。 ●データを判断する。
4	未定義行番号エラー	●GOTO文、GOSUB文の指定行番号がない。	●指定行番号を修正する。
5	引数エラー	●引数を必要とするコマンド、関数において、引数が入力範囲外の場合。	●引数の誤りを修正する。
6	変数エラー	●増設されていないメモリーを使おうとした。 ●同一メモリーを数値変数と文字変数に同時に使おうとした。	●適切にメモリーを増設する。 ●同時に同一メモリーを文字変数、数値変数として使わないようにする。
7	ネスティングエラー	●サブルーチン実行中以外でRETURN文が出てきた場合。 ●FORループ中以外でNEXT文が出てきたり、FOR文に対するNEXT文の変数が異なる場合。 ●サブルーチンのネスティングが８段をこえた場合。 ●FORループのネスティングが４段をこえた場合。	●不必要なRETURN文やNEXT文を取る。 ●サブルーチンやFOR･NEXTループをレベル内にする。
9	オプションエラー	●プリンタやテープレコーダーが接続されていないのにプリントモードで実行したりSAVE等のオプションコマンドを実行した場合。	●プリンタまたはテープレコーダーを接続する。 ●プリントモードを解除する。

プログラムコマンド一覧表

分類	コマンド名	書式	機能
入力文	INPUT	INPUT　変数列	プログラム実行中にキーボード上からデータを入力させます。入力終了までプログラムは停止します。 P−44
	KEY	文字変数＝KEY	プログラム実行中に入力されたキャラクターを読み込み、文字変数に代入する。プログラム実行中に停止はしないため、キー入力がない場合はなにも入りません。 P−45
出力文	PRINT	PRINT　出力制御関数 {；/，} 出力要素 [{；/，}…]	指定された出力要素を出力します。 P−46
	CSR	CSR　n {；/，} （$0 \leq n \leq 11$）	指定された n 桁目から表示します。 P−47
分岐	GOTO	GOTO　{行番号/変数}	指定された行番号へプログラムの進行をジャンプさせます。 P−47
	IF‥{THEN/；}…	IF　比較式 {THEN　行番号/；　コマンド}	比較式が真ならばTHEN以降の行番号へジャンプするか、；以降のコマンドを実行し、偽ならば次の行番号に進みます。 P−48
	GOSUB	GOSUB　{行番号/変数}	指定された行番号のサブルーチンを呼び出し、サブルーチン内のプログラムを実行します。プログラム実行後はRETURN文により再びGOSUB文に戻り次のコマンドに進みます。 P−51
	RETURN	RETURN	サブルーチン内のプログラムの終了を意味し、進行をGOSUB文の直後へ戻します。 P−51
反復	FOR	FOR　$v=e_1$ TO e_2〔STEP e_3〕 ※vは単純数値変数、e_1, e_2, e_3は数式	数値変数 v に対し、初期値を e_1、終値を e_2、刻みを e_3 としたループの開始を宣言します。 ループはFOR文からNEXT文の間で $\left[\frac{e_2-e_1}{e_3}\right]+1$ 回くり返されます。 刻み e_3 が省略された場合は $e_3=1$ と見なされます。 P−49
	NEXT	NEXT　v	FORループの最後を意味し、v に e_3 が加えられ、e_2 を越えてない時は再びループを繰り返し、越えた時はNEXT文の次の行に進みます。 P−49

分　類	コマンド名	書　式	機　能
実行の停止	STOP	STOP	プログラムの実行を一次停止させ、キー入力待ちの状態になります。EXEキーにより実行の継続ができます。 P−54
実行の終了	END	END	プログラムの終了を意味し、実行前の状態に戻ります。EXEキーによる継続はできません。 P−54
データのクリヤー	VAC	VAC	プログラム用の変数データを全てクリヤーします。 P−57
プログラムのリスト	LIST	LIST〔行番号〕	指定された行番号以降のプログラムリストを表示します。 P−55
全プログラムのリスト	LIST A	LIST A	プログラムリストを順次、表示します。 P−55
プログラムの実行	RUN	RUN〔行番号〕	指定された行番号から、プログラムを開始します。 P−54
プログラムの消去	CLEAR	CLEAR	現在指定されているプログラムエリアのプログラムをクリヤーします。 P−58
	CLEAR A	CLEAR A	メモリー内の全てのプログラムをクリヤーします。 P−58
角度単位指定	MODE	MODE $\left\{\begin{matrix}4\\5\\6\end{matrix}\right\}$	三角関数の角度単位を度(4)、ラジアン(5)、グラジアン(6)に指定します。 P−55
フォーマット指定	SET	SET $\left\{\begin{matrix}E\ n\\F\ n\\N\end{matrix}\right\}$ $(0\leqq n\leqq 9)$	表示される数値データの有効桁数または小数以下を指定します。 P−24
文字関数	LEN	LEN(文字変数)	文字変数の大きさを計算します。 P−56
	MID	MID(m〔, n〕)	専用文字変数($)内の第$m$番目から$n$文字を取り出します。 P−56
	VAL	VAL(文字変数)	文字変数内の数字を数値に変換します。 P−57
オプション用	SAVE	SAVE〔"ファイル名"〕	現在指定されているプログラムエリアのプログラムのみテープに記録します。 P−59

分　類	コマンド名	書　式	機　能
オプション用	LOAD	LOAD〔"ファイル名"〕	現在指定されているプログラムエリアにテープからプログラムを呼び戻します。 P－59
	SAVE A	SAVE A〔"ファイル名"〕	全プログラムエリアのプログラムを同時にテープに記録します。 P－59
	LOAD A	LOAD A〔"ファイル名"〕	全プログラムエリアにテープから同時にプログラムを呼び戻します。 P－60
	PUT	PUT〔"ファイル名"〕変数	変数内のデータをテープに記録します。 P－60
	GET	GET〔"ファイル名"〕変数	変数にテープからデータを呼び戻します。 P－60
	VER	VER〔"ファイル名"〕	テープに記録されているプログラムまたはデータが正しく記録されているかチェックします。 P－61

〔　〕内は省略可、｛　｝内はいずれか1つ使用

関数桁容量	入力範囲	答の精度
$\sin x$、$\cos x$、$\tan x$	$\lvert x\rvert < 1440°$ (8πrad、1600gra)	10桁目±1
$\sin^{-1}x$、$\cos^{-1}x$	$\lvert x\rvert \leqq 1$	〃
$\tan^{-1}x$		〃
$\log x$、$\ln x$	$x > 0$	〃
e^x	$x = 1$	〃
$\sqrt{x}$	$x \geqq 0$	〃
x^y ($x\uparrow y$)	$x > 0$	〃

規　格

型　　式：PB-100

基本計算機能：負数、指数、カッコを含む四則計算(加減・乗除の優先順位判別機能つき)

組込関数機能：三角・逆三角関数(角度単位は度・ラジアン・グラジアン)、対数・指数関数、開平、べき乗、整数化、整数部除去、絶対値、符号化、有効桁数指定、小数以下指定、乱数、π

コマンド：INPUT、PRINT、GOTO、FOR・NEXT、IF-THEN、GOSUB、RETURN、STOP、END、RUN、LIST、LIST A、MODE、SET、VAC、CLEAR、CLEAR A、DEFM、SAVE、SAVE A、LOAD、LOAD A、PUT、GET、VER

プログラム関数：KEY、CSR、LEN、MID、VAL

計算範囲：$\pm 1\times10^{-99}\sim\pm 9.999999999\times10^{99}$および0、内部演算は仮数部12桁を使用

プログラム方式：ストアード方式

プログラム言語：BASIC(ベーシック)

ステップ数：最大544ステップ(オプションRAM接続時最大1568ステップ)

組込プログラム数：最大10組(P0～P9)

メモリー数：26～最大94メモリー(オプションRAM接続時最大222メモリー)および専用文字変数($)

スタック数：サブルーチン　8段
FOR・NEXTループ　4段
数　値　6段
演算子　12段

表示方式および内容：仮数部10桁(負符号含む)、または仮数部8桁(負数7桁)＋指数部2桁、EXT、[S]、RUN、WRT、DEG、RAD、GRA、TR、PRT、STOPの各状態表示付

表示素子：12桁ドットマトリクス液晶

主要素子：C-MOS VLSI 他

電　源：リチウム電池(CR-2032)2個使用

消費電力：最大 0.02W

電池寿命(連続使用)：本体のみ　約360時間

オートパワーオフ：約7分

使用温度：0°C～40°C

大きさ・重さ：幅165 奥行71 高さ9.8mm、116g(電池込み)

付属品：ソフトケース、入門書

カシオ計算機株式会社営業本部

東京都新宿区西新宿2—6　新宿住友ビル
(〒160)　☎03-347-4811(代表)

カシオ計算機サービスセンター

拠点	電話	〒	住所
旭川	0166-23-8580	〒070	旭川市七条通り8丁目
札幌	011-231-2343	〒060	札幌市中央区南一条西12丁目
釧路	0154-24-8575	〒085	釧路市光陽町6—7
青森	0177-22-7466	〒030	青森市勝田2—1—12
秋田	0188-63-7690	〒010	秋田市山王2—1—40
盛岡	0196-24-2502	〒020	盛岡市本町通り3—19—6
仙台	0222-27-1404	〒980	仙台市一番町2—3—32
山形	0236-42-8018	〒990	山形市あこや町3—12—9
郡山	0249-33-5172	〒963	福島県郡山市香久池2—16—6
宇都宮	0286-34-0395	〒320	宇都宮市西大寛2—1—3
前橋	0272-53-3000	〒371	前橋市元総社町92—5
水戸	0292-25-6985	〒310	水戸市中央1—2—20
埼玉	0486-66-8567	〒330	大宮市大成町1—181
千葉	0472-43-1751	〒280	千葉市登戸町2—276
東京	03—862-4141	〒101	千代田区神田佐久間町2—23
東京	03—583-4111	〒106	港区六本木2—3—6
城南	03—787-3721	〒145	大田区上池台1—1—6
城西	03—376-3221	〒160	新宿区西新宿4—2—18
多摩	0425-23-3531	〒190	立川市錦町3—2—25
横浜	045-211-0821	〒231	横浜市中区弁天通り6—85
平塚	0463-23-2611	〒254	平塚市新宿1196
新潟	0252-41-4105	〒950	新潟市米山3—1—5
長野	0262-28-9360	〒380	長野市岡田町30—20
甲府	0552-37-6371	〒400	甲府市城東2—22—11
沼津	0559-22-8928	〒410	沼津市高島町8—11
静岡	0542-81-8085	〒420	静岡市西中原1—4—35
浜松	0534-64-1658	〒435	浜松市西塚町324
豊橋	0532-53-2515	〒440	豊橋市魚町5
名古屋	052-263-0454	〒460	名古屋市中区栄4—6—15
岐阜	0582-62-0145	〒500	岐阜市鷹見町8
三重	0592-27-5066	〒514	津市鳥居町191
富山	0764-22-2251	〒930	富山市白銀町2—1
金沢	0762-37-8511	〒920	金沢市諸江町下丁93—1
京都	075-351-1161	〒600	京都市下京区五条通り堀川東入ル
大阪	06—362-8181	〒530	大阪市北区南森町2—1—20
奈良	0742-24-3811	〒630	奈良市西木辻町200—62
和歌山	0734-31-7807	〒640	和歌山市九家の丁5
神戸	078-392-4123	〒650	神戸市中央区北長狭通り4—4—18
岡山	0862-41-8471	〒700	岡山市西古松406—2
松江	0852-25-1311	〒690	松江市嫁島町173
福山	0849-24-2830	〒720	福山市南本庄町2—130
広島	082-263-1090	〒730	広島市稲荷町4—1
山口	0835-22-6164	〒747	防府市戎町1—10—16
高松	0878-62-5240	〒760	高松市亀岡町9—16
松山	0899-45-2234	〒790	松山市平和通り1—1—5
福岡	092-411-2684	〒812	福岡市博多区博多駅南1—2—15
長崎	0958-61-8084	〒852	長崎市宝栄町2—26
熊本	0963-67-0650	〒862	熊本市健軍4—1—5
鹿児島	0992-56-3575	〒890	鹿児島市上荒田町30—18